Hindawi
Mobile Information Systems
Volume 2021, Article ID 9993946, 13 pages
https://doi.org/10.1155/2021/9993946

Research Article

Smart Application Division and Time Allocation Policy for Computational Offloading in Wireless Powered Mobile Edge Computing

Abdullah Numani,[1] **Zaiwar Ali**,[1] **Ziaul Haq Abbas**,[2] **Ghulam Abbas**,[3] **Thar Baker**,[4] **and Dhiya Al-Jumeily**[5]

[1]*Telecommunications and Networking (TeleCoN) Research Lab, GIK Institute of Engineering Sciences and Technology, Topi, Swabi 23640, Pakistan*
[2]*Faculty of Electrical Engineering, GIK Institute of Engineering Sciences and Technology, Topi, Swabi 23640, Pakistan*
[3]*Faculty of Computer Sciences and Engineering, GIK Institute of Engineering Sciences and Technology, Topi, Swabi 23640, Pakistan*
[4]*Department of Computer Science, University of Sharjah, Sharjah, UAE*
[5]*Department of Computer Science, Liverpool John Moores University, Liverpool, UK*

Correspondence should be addressed to Thar Baker; tshamsa@sharjah.ac.ae

Received 3 April 2021; Revised 29 May 2021; Accepted 24 June 2021; Published 2 July 2021

Academic Editor: Yugen Yi

Copyright © 2021 Abdullah Numani et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Limited battery life and poor computational resources of mobile terminals are challenging problems for the present and future computation-intensive mobile applications. Wireless powered mobile edge computing is one of the solutions, in which wireless energy transfer technology and cloud server's capabilities are brought to the edge of cellular networks. In wireless powered mobile edge computing systems, the mobile terminals charge their batteries through radio frequency signals and offload their applications to the nearby hybrid access point in the same time slot to minimize their energy consumption and ensure uninterrupted connectivity with hybrid access point. However, the smart division of application into k subtasks as well as intelligent partitioning of time slot for harvesting energy and offloading data is a complex problem. In this paper, we propose a novel deep-learning-based offloading and time allocation policy (DOTP) for training a deep neural network that divides the computation application into optimal number of subtasks, decides for the subtasks to be offloaded or executed locally (offloading policy), and divides the time slot for data offloading and energy harvesting (time allocation policy). DOTP takes into account the current battery level, energy consumption, and time delay of mobile terminal. A comprehensive cost function is formulated, which uses all the aforementioned metrics to calculate the cost for all k number of subtasks. We propose an algorithm that selects the optimal number of subtasks, partial offloading policy, and time allocation policy to generate a huge dataset for training a deep neural network and hence avoid huge computational overhead in partial offloading. Simulation results are compared with the benchmark schemes of total offloading, local execution, and partial offloading. It is evident from the results that the proposed algorithm outperforms the other schemes in terms of battery life, time delay, and energy consumption, with 75% accuracy of the trained deep neural network. The achieved decrease in total energy consumption of mobile terminal through DOTP is 45.74%, 36.69%, and 30.59% as compared to total offloading, partial offloading, and local offloading schemes, respectively.

1. Introduction

Due to the advancements in cellular technologies and applications of Internet of things (IoT), our daily life activities, such as smart healthcare, smart homes, and smart driving, are becoming dependent on mobile terminals (MTs) [1, 2]. The aforementioned applications are computation-intensive and energy-hungry and require large storage and faster execution [3]. To meet these requirements, an attempt is the cloud computing system that contains centralized servers having higher processing powers and storage capabilities [4, 5]. However, the delay-sensitive applications such as online simulators, live streaming, and online games cannot be executed through cloud servers due to the inherited problem of high latency caused by long distances of cloud servers from the MTs [6, 7]. The problems of high latency and centralized architecture can be overcome to a certain limit through fog nodes closer to the request generators in fog computing. However, there are challenges of tasks offloading, resource distribution, and degraded performance of network links [8].

An alternative for the delay-sensitive applications is mobile edge computing (MEC) that offers services similar to that of cloud computing in a distributed manner and can be called the cloud computing at the edge of cellular networks. In the MEC environment, an MT can offload its computational applications to the nearby mobile edge server (MES) available with the lowest delays as compared to the cloud servers. The challenging problems of MTs, that is, limited computational power, storage, and higher time delays in execution, are entirely dependent on the battery life of MT. Unfortunately, the MTs have limited battery life that leads to the compromised performance [9]. To ensure the connectivity of MT with a nearby base station (BS), the battery of MT needs regular charging that requires more hardware and physical cost. The technology of wireless energy transfer (WET) is explained in [10], in which the battery of MTs can be charged wirelessly through radio frequency (RF) signals. However, energy harvesting from RF signals transmitted for information exchange leads to security issues at physical layer. Hence, either the deployment of multiple antennas at MT and exploiting diversity techniques or energy transmitter (ET) at the BS is preferred [11]. Recently, incorporation of WET technology in MEC system is a hot area of research. A model that utilizes the advantages of both WET technology and MEC is called wireless powered MEC (WP-MEC) [12]. The use of such model can facilitate the MT with the harvesting of energy to improve the battery life and with the computational offloading to enhance the computational capability of MT simultaneously.

There are two main types of computational offloading. In coarse-grained computational offloading, the whole application is either executed locally at the MT or offloaded to the remote MES [13]. Meanwhile, in fine-grained computational offloading [14], the whole application of size M is first divided into a k number of subtasks and can be offloaded with 2^k possible policies. For example, if an application of 10 GB is divided into 4 subtasks, then these 4 subtasks can be offloaded according to $2^4 = 16$ offloading policies as shown in Table 1.

Number of subtasks per application and selection of offloading policy mainly contribute to energy consumption and time delay of MT. Therefore, it is important to divide the application in an optimal number of subtasks and select an offloading policy that has minimum cost.

In WP-MEC system, an MT offloads its computational application to the hybrid access point (HAP) and harvests energy from RF signals of energy transmitter (ET). The MT harvests energy in some part of time slot and offloads data in the remaining time of the same time slot. Therefore, the intelligent division of time slot into fractions for energy harvesting and data offloading is a crucial point of consideration, known as time allocation policy. The possible number of time allocation policies is dependent on resolution of time, τ_r, and is equal to $(1/\tau_r) + 1$. Decrease in τ_r increases the number of time allocation policies. We can find the cost of all possible policies and then select the optimal one that has minimum cost. For example, the possible time allocation policies for $\tau_r = 0.2$ are given in Table 2. Both the battery life and computation power can be improved through the optimal selection of number of subtasks per application and time allocation policy for energy harvesting and data offloading. The time allocation policy is investigated through the Lyapunov function optimization technique in [15] for WP-MEC system. However, such numerical solutions suffer from huge computational overhead, which increases the complexity of algorithm and time delay of execution. To avoid the problem of huge computational overhead in such offloading, the emerging artificial intelligence (AI) concepts can be used.

In this paper, we propose a deep learning approach to find the best time allocation policy and partial offloading policy with optimal number of subtasks per application. Our proposed cost function depends on time delay, energy consumption, and current battery level of the MT. Initially, if the battery level is 100%, our cost function focuses on time delay of application execution. The cost function gives more importance to the energy consumption when the battery is low. The considered WP-MEC system consists of an HAP (containing an ET, MES, and an AP) and MTs. Due to the size limitation, MT is considered to have one antenna that can be used for both energy harvesting and computational offloading. In contrast, the larger size of HAP allows separate antennas for transmitting energy and data communication, as shown in Figure 1. The target of the proposed approach is to utilize the WET and partial offloading technique with optimal number of subtasks per application to improve the battery life of MT and make the execution of applications faster. Novelty and contributions of the proposed work are outlined as follows:

Table 1: Offloading policies for an application divided into 4 subtasks.

	Offload to HAP = 1, execute on MT = 0			
Policy	Subtask 1	Subtask 2	Subtask 3	Subtask 4
1	0	0	0	0
2	0	0	0	1
3	0	0	1	0
4	0	0	1	1
5	0	1	0	0
6	0	1	0	1
7	0	1	1	0
8	0	1	1	1
⋮	⋮	⋮	⋮	⋮
16	1	1	1	1

Table 2: Time allocation policies for time resolution of $\tau_r = 0.2$.

Possible options	Time fraction for harvesting energy (t_0)	Time fraction for data offloading (t_1)
Policy 1	0	1
Policy 2	0.2	0.8
Policy 3	0.4	0.6
Policy 4	0.6	0.4
Policy 5	0.8	0.2
Policy 6	1	0

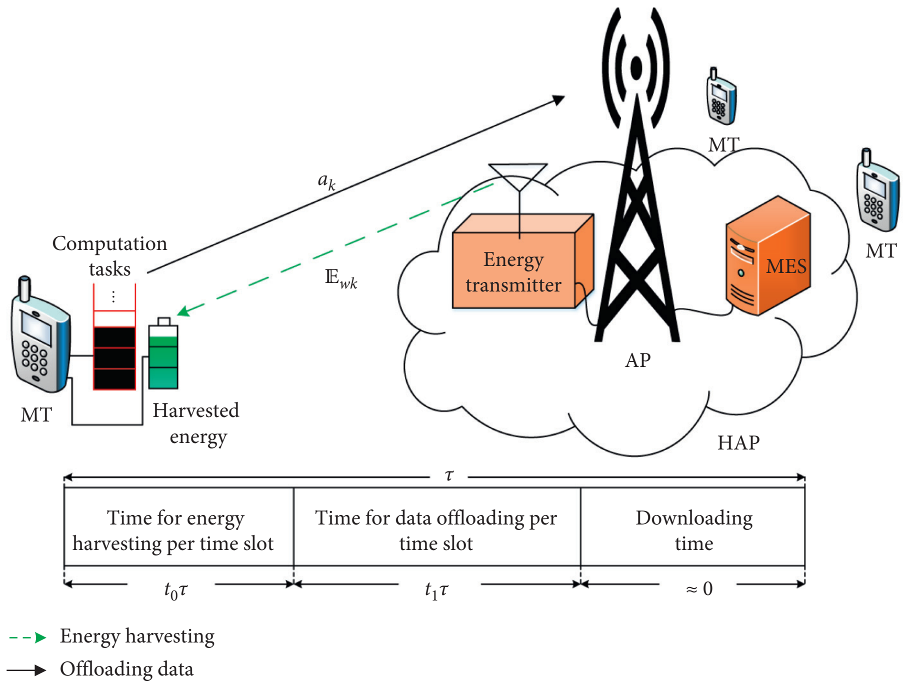

Figure 1: Wireless powered mobile edge computing system.

(i) We find the time allocation policy for energy harvesting and data offloading, partial offloading policy, and optimal number of subtasks per application, simultaneously, through deep learning approach in WP-MEC system to improve the battery life of MT and execution delay.

(ii) We formulate a cost function which depends on energy consumption, time delay, and current battery life of MT. There are two main parts of the cost function: energy consumption part and time delay part. The minimum cost value means that the energy consumption and time delay will be minimum; however, our cost function updates itself through weighting coefficients dependent on battery level, which decide the contribution of time delay part or energy consumption part in the cost function.

(iii) Simulation results show that our proposed algorithm prolongs the battery life and ensures faster execution and minimum energy consumption.

The remainder of the paper is organized as follows: Section 2 provides the related work of the proposed technique. Section 3 presents the mathematical model and the proposed algorithm. In Section 4, the numerical simulations and results are discussed. Section 5 concludes the paper and suggests some future directions.

2. Related Work

For the improvement of MT's battery life and ensured connectivity with HAP during partial offloading technique, the time slot for data offloading is further divided into two parts: (a) the time fraction in which the MT will charge its battery through RF signal and (b) the time fraction in which the MT will offload data to the HAP. This division of time slot, referred to as time allocation policy, is discussed in [15, 16]. The problem of time allocation policy (for power transfer and data offloading) and offloading mode selection (for offloading or local execution of an application) is studied in [12], and a total computational offloading technique for all energy harvesting MTs is proposed. Based on computation rate, a maximization problem is studied in [17] for the decrease in propagation loss that severely affects harvested energy and computation performance of MT. For the joint optimization of time allocation among MTs, CPU frequencies, transmitted energy at the AP, and number of offloaded bits, a multiuser WP-MEC is considered in [18]. The authors in [15] propose an algorithm for the improvement of computation rate by joint consideration of computational offloading and time allocation policies in a WP-MEC system with one MT. The enhancement in computational performance of active MTs by the user cooperation technique is investigated in [19], where the inactive MTs use their harvested energy for the help of active ones. For the computational offloading policy, the authors in [19] consider frequency division multiple access (FDMA) to improve computation rate. The maximum-minimum energy efficiency optimization problem (MMEP) with joint optimization of energy consumption, time slots for computational offloading and energy transfer, and transmitted power at HAP in WP-MEC system are focused on in [20], and, by the application of block coordinate descent (BCD) and fractional programming theory, the authors present algorithms with lower complexity. In [21], the computational energy efficiency of entire system is improved by the joint consideration of optimal allocation for MT's transmitted power, CPU frequency, and time for transmission in WP-MEC system. In [22], the maximization problem of system energy efficiency by the joint consideration of optimal time allocation, local computing capacity, energy consumption, and application offloading is discussed. In [23], the authors consider a stochastic method for battery management and resource allocation decisions in a time slot and propose an algorithm derived from Lyapunov optimization technique. In [24], the authors utilize the Lyapunov optimization for the evaluation of tradeoff between delay and energy efficiency of a multiuser WP-MEC system. In all the above-mentioned work, the improvement in computation rate is mainly focused. However, the battery life of MT is ignored and the number of subtasks in which an application should be divided for partial offloading is assumed to be a fixed value.

In most studies on WP-MEC, the investigation of time allocation and resource allocation policies is based on the numerical optimization techniques such as Lyapunov and Lagrange multipliers. However, for the complex optimization problems of allocating resources in WP-MEC, the said techniques do not guarantee global minima/maxima, and thus the performance of algorithms based on these methods becomes poor. Additionally, these approaches make use of long calculations and thus involve larger computational overheads that affect the time delays for computations and energy consumption of MTs. The greater amount of energy consumption badly affects the battery life of MTs and consequently the connectivity of MTs with HAP becomes limited. An alternate solution for fast computation of application division in subtasks and provision of optimal decisions for time allocation and resource allocation policies with minimum energy consumption and improved battery life is required. This requirement leads to the deep learning approach that decreases computational complexity by learning from predefined dataset training.

Little work is done for optimal time allocation and offloading policies in WP-MEC systems using deep learning approach. In [25], the authors investigate deep reinforcement learning for online offloading to reduce the computational complexity of large-size networks for offloading decisions and resource allocations, while ignoring the cost function. The main focus of the authors in [25] is on the minimization of execution latency, while battery life of MT, energy consumption, and optimal division of application are not taken into account. In [26, 27], the authors consider metrics such as subtask energy consumption, MT's remaining energy, computational load, and network conditions and propose a deep-learning-based offloading policy. However, their work ignores the energy harvesting capability of HAP and considers only MES at the edge of cellular network. A comparison table of the work related to deep-learning-based offloading and time allocation policy (DOTP) is given in Table 3.

From the study of related work on binary offloading policies in WP-MEC systems, we find two options for an application execution. (1) If an application is locally executed at the MT, it can reduce energy consumption; however, due to limited computational resources, the time delay will be higher. (2) If all the subtasks of an application are remotely executed on MES, the computational time delay will be smaller as the MES is assumed to have sufficient computational resources; however, in offloading data, the energy consumption of the MT will be higher. The third option is the partial offloading policy, that is, to divide the application into a number of subtasks. Then, some of the subtasks will be locally executed, while some subtasks are offloaded to the MES for remote execution. In this case, the option for subtask to be executed either locally on MT or remotely on the MES is dependent on the cost calculated through the cost function. This paper focuses on a deep-learning-based approach to find the optimal number of subtasks (optimal in terms of cost) in which we should divide the application and then apply the partial offloading policy. In the offloading option of a subtask, the time allocation policy is applied to find the time fraction for MT's energy harvesting and data offloading per time slot. The proposed cost function jointly considers the remaining battery level, time delay, and energy consumption of MT in the cost calculation for an application. Simulation results show that the proposed DOTP technique performs better than the benchmark schemes in terms of battery life, time delay, and energy consumption.

3. System Model

In our WP-MEC system model, we consider an HAP communicating with single MT. From implementation point of view, the HAP is assumed to have two antennas used in a time division duplex (TDD) manner for energy transmission and data communication separately, in the same frequency band. Meanwhile, the MT is considered to have single antenna; therefore, it utilizes the protocol of first-harvest-then-offload through the same antenna on downlink (DL) and uplink (UL), respectively. During the remote execution, MT first harvests energy for some part of the time slot and then transmits its subtask for execution to the MES in the rest of time slot. Moreover, the HAP is assumed to have sufficient resources of energy and processing, while the

Table 3: Comparison of related works with the proposed DOTP technique in WP-MEC system.

Technique	Considers battery life?	Considers optimal number of subtasks per application?	Considers energy consumption?	Considers time delay?	Considers energy harvesting (WET technology)?	Considers deep learning approach?
Lyapunov optimization [23] (2016)	Yes	No	Yes	No	Yes	No
Wang et al. [18]	No	No	Yes	Yes	Yes	No
Mao et al. [20]	No	No	Yes	No	Yes	No
ADMM [12] (2018)	No	No	Yes	Yes	Yes	No
Zhou et al. [17]	No	No	Yes	No	Yes	No
Lagrange multipliers [19] (2018)	No	No	Yes	No	Yes	No
OSRM [15] (2019)	No	No	Yes	No	Yes	No
Mao et al. [24]	No	No	Yes	Yes	Yes	No
DROO [25] (2019)	No	No	No	Yes	Yes	Yes
EEDOS [26] (2019)	No	Yes	Yes	Yes	No	Yes
SEEM [22] (2020)	Yes	No	Yes	No	Yes	No
MADS [21] (2020)	No	No	Yes	No	Yes	No
JTAOP [16] (2021)	No	No	Yes	No	Yes	No
CEDOT [27] (2021)	No	Yes	Yes	Yes	No	Yes
DOTP (our proposed technique)	Yes	Yes	Yes	Yes	Yes	Yes

MT is considered to have limited computational resources and battery. Thus, the MT will use the energy transmitted by HAP to prolong its battery life for offloading and local execution of subtasks. In partial offloading, some of the components are offloaded to MES for execution. We divided a time slot for offloading in two parts, called time allocation policy. In the first part, the MT harvests energy and in the second part it offloads its data to the MES. The downloading time is small as compared to the transmitting time; therefore, the downloading time is assumed to be negligible. The time allocation policy is denoted by $t = \{t_0, t_1\}$, where t_0 is the percent time of a single time slot for which the MT will harvest energy from HAP and t_1 is the percent time of the same time slot in which the MT will offload its subtask to the MES. We introduce a binary variable B_k for subtask k which defines the decision of local execution or remote execution. $B_k = 0$ indicates the local execution of a subtask k on MT and $B_k = 1$ indicates the remote execution of a subtask k at the MES. A detailed list of notations used in this paper is given in Table 4. Two models are considered, namely, local execution model and remote execution model.

3.1. Local Execution.

For local execution of subtask k, that is, $B_k = 0$, the time delay of local execution, D_{lk}, is calculated as

$$D_{lk} = \frac{a_k C}{\nu_{mk}}, \tag{1}$$

where ν_{mk} is the CPU frequency of MT for subtask k, C is the number of CPU cycles used for the execution of one byte of data, and a_k is the input data of subtask k, executed locally at the MT. The energy consumption of MT for a single subtask k in local execution is formulated as

$$\mathbb{E}_{lk} = \eta \nu_{mk}^3 D_{lk}, \tag{2}$$

where η, a dimensionless quantity, represents the effective switching capacitance factor that is dependent on the architecture of chip [28]. The product $\eta\nu_{mk}$ shows the computation power of MT. The total cost for subtask k executed locally at the MT is defined as the summation of normalized energy consumption and normalized time delay for subtask k and is given by

$$\lambda_{lk} = \varepsilon \mathbb{E}_{lk} + \psi D_{lk}, \tag{3}$$

where ε and ψ are the unit balancing coefficients defined, respectively, as

$$\begin{aligned} \varepsilon &= \frac{\sigma_1}{\mathbb{E}_{\max}}, \\ \psi &= \frac{\sigma_2}{D_{\max}}. \end{aligned} \tag{4}$$

For the whole application, $\mathbb{E}_{\max}$ and $D_{\max}$ are the maximum energy consumption and maximum time delay, respectively, having fixed values. σ_1 and σ_2 are the weighting coefficients that are dependent on the current level of MT's battery. These coefficients are used for providing priority to either data offloading or energy harvesting of MT. The cost function checks the current level of battery and decides the time fraction for energy harvesting. Initially, when the battery level is high, the time fraction for energy harvesting is low and the time fraction for offloading is high. However,

Table 4: Notations.

Notations	Meaning
a_k	Input data of subtask k
a_{k+1}	Output data of subtask k
α_u, α_d	Uplink and downlink small scale fading channel power gains, respectively
B_k	Boolean variable for offloading option of subtask k
b^*	Optimal offloading policy
$\mathcal{B}_i$	Current available energy of MT's battery
$\mathcal{B}_{i+1}$	Energy of MT's battery for coming time slot
$\mathcal{B}_{\%}$	Percent current level of MT's battery
β	Available bandwidth
C	CPU cycles to execute 1 byte of data
D_{dk}, D_{uk}	MT's downlink reception and uplink transmission time delays, respectively
D_{ek}, D_k	Processing and total time delays for subtask k, respectively
$D_{\max}$	Maximum time delay for single application
D_{lk}, D_{ok}	Time delays for local and remote execution of subtask k, respectively
$\mathbb{E}_{lk}$, $\mathbb{E}_{ok}$	Local and remote execution energy consumption for subtask k, respectively
$\mathbb{E}_{\max}$	Maximum energy consumption for single application
$\mathbb{E}_{wk}$	Harvested energy of MT during subtask k
η	Effective switching capacitance factor
ε and ψ	Unit balancing coefficients
G_k	Channel power gain for subtask k
γ_k^{ul}, γ_k^{dl}	MT's uplink and downlink channel data rates, respectively
k	Number of subtasks
λ	Cost function
λ_{lk}, λ_{ok}	Cost functions for local execution and remote execution of subtask k, respectively
λ^*	Optimal cost
M	Size of application
N_0	Noise spectral density
n	Number of subtasks per application
n^*	Optimal number of subtasks
ν_h, ν_{mk}	CPU frequencies of MES and MT, respectively
OP	Matrix of all possible offloading policies
ϕ	Path loss exponent
$\mathbb{P}_{hk}$, $\mathbb{P}_{mk}$	Transmitted powers of HAP and MT, respectively
$\mathbb{P}_x$	Downlink power of HAP
ρ	Path loss constant
s	Distance between MT and HAP
s_0	Reference distance
σ_1 and σ_2	Weighting coefficients
t_0, t_1	MT's time fractions for harvesting energy and data offloading, respectively
t_0^*, t_1^*	Optimal time fractions for energy harvesting and data offloading, respectively
τ_r	Time resolution

when the battery level decreases and falls below a threshold value, our cost function increases the time fraction for energy harvesting to its maximum value.

3.2. Remote Execution. For remote execution, that is, $B_k = 1$, the subtask k is offloaded to the HAP to utilize the processing power of MES. The processing power of MES and transmission power of HAP are assumed to be sufficient. Consequently, the delay of downloading from HAP to the MT is taken as zero and the total time is divided into two parts only, that is, $t_0 + t_1 \approx 1$ [22]. The UL channel data rate from MT to HAP, γ_k^{ul}, can be obtained using the Shannon-Hartley theorem as follows:

$$\gamma_k^{ul} = \beta \log_2\left(1 + \frac{G_k \mathbb{P}_{mk}}{\beta N_0}\right), \tag{5}$$

where $\mathbb{P}_{mk}$ denotes the transmitted power of MT for subtask k. For the UL, noise spectral density is represented by N_0 and β is the available bandwidth. The wireless channel between MT and HAP is considered to be independent and identically distributed block fading channel. α_u represents the UL small scale fading channel power gain. The channel power gain from MT to HAP is calculated as $G_k = \rho\alpha_u (s_0/s)^{\phi}$, where ρ is the path loss constant, s_0 is the reference distance, ϕ is the path loss exponent, and s is the distance between MT and HAP [29]. In a similar way, the DL channel data rate from HAP to the MT, γ_k^{dl}, can be formulated as

$$\gamma_k^{dl} = \beta \log_2\left(1 + \frac{G_k \mathbb{P}_{hk}}{\beta N_0}\right), \tag{6}$$

where $\mathbb{P}_{hk}$ represents the transmitted power of HAP for subtask k.

For calculating the time delay for remote execution, the UL time for data transmission, the DL time for data reception, and the execution time for data at MES are considered. The UL time delay in transmission of data from MT to the HAP, D_{uk}, is given as

$$D_{uk} = \frac{a_k}{\gamma_k^{ul}}. \tag{7}$$

In a similar way, the DL time delay in reception of data from HAP to the MT, D_{dk}, is formulated as

$$D_{dk} = \frac{a_{k+1}}{\gamma_k^{dl}}, \tag{8}$$

where a_{k+1} is the output data of subtask k and the input data of subtask $k+1$. The execution time delay for data a_k at the MES, D_{ek}, is calculated as

$$D_{ek} = \frac{a_k}{\nu_h}, \tag{9}$$

where ν_h is the CPU frequency of MES. The offloading time delay for a subtask k, D_{ok}, is obtained as

$$D_{ok} = t_1 \left(D_{uk} + D_{dk} + D_{ek}\right). \tag{10}$$

The MES is considered to have high computation power due to a large number of CPUs in it. Thus, due to high CPU frequency of MES, the execution time delay, D_{ek}, may be ignored. Moreover, the output data for k subtask, a_{k+1}, is very small as compared to the input data, a_k, and hence the DL time delay, D_{dk}, can be considered as approximately zero. Consequently, the offloading time delay of subtask k, D_{ok}, can be approximated as

$$D_{ok} \approx t_1 D_{uk}. \tag{11}$$

The MT's offloading energy consumption of subtask k, $\mathbb{E}_{ok}$, in remote execution model is formulated as

$$\mathbb{E}_{ok} = D_{ok}\mathbb{P}_{mk}. \tag{12}$$

The total cost for subtask k, executed remotely at the MES, is given by

$$\lambda_{ok} = \varepsilon \mathbb{E}_{ok} + \psi D_{ok}, \tag{13}$$

where ε and ψ are the unit balancing coefficients, as in (3).

The total time delay of a subtask k, D_k, can be defined as

$$D_k = \begin{cases} D_{ok}, & B_k = 1, \\ D_{lk}, & B_k = 0. \end{cases} \tag{14}$$

The cost of subtask k either locally executed on MT or remotely executed at the MES is formulated as

$$\lambda_k = \begin{cases} \lambda_{ok}, & B_k = 1, \\ \lambda_{lk}, & B_k = 0. \end{cases} \tag{15}$$

The total time delay for an application containing n number of subtasks, D_a, can be obtained as

$$D_a = \sum_{k=1}^{n} D_{ok}B_k + \sum_{k=1}^{n} D_{lk}\left(1 - B_k\right). \tag{16}$$

In this paper, it is considered that the MT has a chargeable battery, which can provide and store energy for the consumption of MT and can harvest energy from HAP. The harvested energy of MT is calculated as

$$\mathbb{E}_{wk} = t_0 D_{ok}\mu\mathbb{P}_x\alpha_d, \tag{17}$$

where $\mu = \begin{bmatrix} 0 & 1 \end{bmatrix}$ is a constant for energy harvesting and describes how efficiently the MT receives the transmitted energy of HAP [6]. For broadcasting RF signals to the MTs, $\mathbb{P}_x$ is the transmitted power of HAP. α_d represents the DL channel power gain. For the purpose of simplicity, it is assumed that α_d is equal to the UL channel power gain G_k. For subtask k, the total energy consumption of MT, $\mathbb{E}_k$, is written as

$$\mathbb{E}_k = \begin{cases} \mathbb{E}_{ok}, & B_k = 1, \\ \mathbb{E}_{lk}, & B_k = 0. \end{cases} \tag{18}$$

The current available energy of MT's battery is denoted by $\mathscr{B}_i$ and satisfies the condition

$$\mathbb{E}_k \leq \mathscr{B}_i + \mathbb{E}_{wk}. \tag{19}$$

The available energy of MT's battery is updated as follows:

$$\mathscr{B}_{i+1} = \mathscr{B}_i - \mathbb{E}_k + \mathbb{E}_{wk}, \tag{20}$$

where $\mathscr{B}_{i+1}$ is the energy of MT's battery for coming time slot. The total energy consumption of MT for single application, containing n number of subtasks, $\mathbb{E}_a$, is defined as

$$\mathbb{E}_a = \sum_{k=1}^{n} \mathbb{E}_{ok}B_k + \sum_{k=1}^{n} \mathbb{E}_{lk}\left(1 - B_k\right). \tag{21}$$

For the comparison of different techniques, a cost function is modeled as

$$\lambda = \sum_{k=1}^{n} \lambda_{ok}B_k + \sum_{k=1}^{n} \lambda_{lk}\left(1 - B_k\right), \tag{22}$$

where λ is the cost for a single application execution.

Initially, if the battery level is 100%, the energy consumption weight should be less than the weight of time delay because, in that case, faster execution is more important. The energy consumption part in the cost function becomes more important when the battery of MT is low. Therefore, we calculate these weighting coefficients by using the current battery level of MT as

$$\begin{aligned} \sigma_1 &= \frac{\mathscr{B}_{\%}}{\mathbb{E}_{\max}}, \\ \sigma_2 &= 1 - \sigma_1, \end{aligned} \tag{23}$$

where $\mathscr{B}_{\%}$ represents the percent current level of battery.

3.3. Proposed DOTP Algorithm. The number of possible values for energy harvesting time t_0 is represented by Z and is given as $Z = (1/\tau_r) + 1$. In the proposed DOTP technique, described in Algorithm 1, we calculate the local and remote costs for all possible number of subtasks per application, $k \in \{1, 2, 3, \ldots, n\}$, number of possible values for $t_0 \in \{0{:}\ \tau_r{:}\ 1\}$, that is Z, and offloading policies (2^n). Thus, the complexity of algorithm is $O(n \times Z \times 2^n)$. **OP** is a matrix of all possible offloading policies and has the order of $(n \times 2^n)$. For subtask k, local and remote costs are calculated using (3) and (13), respectively. With optimal number of subtasks, n^*, optimal time allocation policy, $t_0^* \in \{t_{01}^*, t_{02}^*, t_{03}^*, t_{04}^*, t_{05}^*, t_{06}^*, t_{07}^*, t_{08}^*, t_{09}^*\}$, and optimal offloading policy, b^*, the minimum cost is denoted by λ^* in Algorithm 1, where t_{01}^* is the harvesting time for subtask 1, t_{02}^* is the harvesting time for subtask 2, and so on.

Due to huge computational overhead of finding λ^*, n^*, t_0^*, and b^*, the service delay and energy consumption of MTs increase. Computations of these values are also explained through the flowchart in Figure 2. With the help of cost function, we generate a training dataset and train a DNN to reduce the computational overhead, energy consumption, and service delay of MTs.

The input layer of our DNN takes the size of application (M), number of subtasks per application (n), distance of MT from HAP (s), CPU frequency (ν_m), transmitted power ($\mathbb{P}_m$), and the available battery level of MT ($\mathscr{B}_i$), while the output layer gives the optimal number of subtasks (n^*), partial offloading policy (b^*), and the optimal time for harvesting energy (t_0^*). For the two hidden layers and the output layer, the rectified linear unit (ReLU) and Softmax activation functions are used, respectively.

We calculate the minimum cost and store the corresponding inputs and outputs as a training dataset. After obtaining a training dataset of size 5000, these data are divided into training dataset (70% of training dataset), validation dataset (15% of the training dataset), and test dataset (15% of the training dataset). The DNN is trained on training dataset and then its performance is validated through validation dataset. When the DNN is trained, the accuracy of DNN is checked through test dataset. The results of our DOTP algorithm are compared with the exiting benchmark schemes, that is, local execution, total offloading, and partial offloading.

4. Simulation and Results

The simulations environment used was MATLAB (R2019a) and the processor was Intel Core i7 with clock speed of 3.4 GHz. An application can be divided into more than 10 subtasks; however, for simulation purpose, we have assumed that an application can be divided into a maximum of 10 subtasks. For time allocation policy, the time resolution is assumed to be 0.01. Practically, the distance for WET is up to 12 m. However, to check the mobility effect of MT in a more comprehensive way, the maximum distance of MT from HAP is assumed to be 200 m. The reference distance of MT from HAP is assumed to be 1 m. The experimentation is performed on the basis of randomly generated data to show the stochastic nature of the problem and have more realistic results. The randomly generated input data for our graphs are the CPU frequency of MT, transmitted power of MT, distance of MT from HAP, and the application size. The advanced smart phones have the heterogeneous frequency capability and use different frequencies in different scenarios [30]. Thus, the CPU frequency of MT is taken randomly from a uniform distribution in the range of $[\,0.1\ \ 1\,]$ GHz. Similarly, the transmitted power of smartphones is variable and changes as the distance of MT from HAP varies. This stochastic nature of transmitted power of MT is incorporated by considering it as a uniform distribution in $[\,0.8\ \ 1.5\,]$ Watts. Different applications may have different sizes and that is why we have considered the size of application to be randomly selected from a uniform distribution in $[\,0.1\ \ 1\,]$ G bits. To incorporate the mobility effect of MT in the simulation, the distance of MT from HAP is selected randomly from a uniform distribution in the range of $[\,3\ \ 200\,]$ m.

CPU cycles of MT to execute one byte of data, C, are taken as 737.5 cycles/byte, each application is divided into n number of subtasks, and the value of n lies between 1 and 10. The value of effective switching capacitance factor, η, is taken as 10^{-25}, MT's efficiency of energy harvesting, μ, is taken as 0.8, path loss constant, ρ, has the value of -30 dB, path loss exponent $\phi \geq 2$, and reference distance of MT from HAP, s_0, is 1 m. The available bandwidth, β, and noise spectral density, N_0, are 0.5 MHz and -174 dBm/Hz, respectively. The trained DNN has 32 neurons in the input layer (one for MT's currently available battery level, $\mathscr{B}_i$, one for application size, M, ten for MT's CPU frequencies, ν_m, ten for MT's transmitted powers, $\mathbb{P}_m$, and ten for the distances of MT from HAP, s). The simulation parameters are given in Table 5. The number of subtasks in which an application of size M will be divided is taken from 1 to 10. Therefore, in the output layer, 10 neurons are specified for the number of subtasks per application. In the partial offloading policy, as considered in this paper, a maximum of 9 subtasks out of 10 can be offloaded to the remote MES. Thus, there are 9 neurons for the optimal time allocation policy, t_0^*. In one neuron at the output layer, the optimal partial offloading policy, b^*, is stored. Thus, the total number of neurons in the output layer of DNN is 20. There are two hidden layers and 100 neurons in each hidden layer. The accuracy of trained DNN achieved is 75%. The detailed architecture of DNN is shown in Figure 3.

The benchmark techniques considered for comparison of simulation results in this paper are described as follows:

(i) Total offloading technique [18]: in total offloading, no policy for offloading decisions is considered and the whole application is offloaded to the MES.

(ii) Local execution [19]: in local execution, the whole application is executed locally on MT and there is no offloading to MES.

(iii) Partial offloading technique [14]: in partial offloading, the application is divided into a number of

```
**Input:** $a \in [a_1, a_2, \ldots, a_n], \nu_m \in [\nu_{\min}, \nu_{\max}], \mathbb{P}_m \in [\mathbb{P}_{\min}, \mathbb{P}_{\max}], s \in [s_{\min}, s_{\max}], \mathcal{B}_i$
**Output:** $\{n^*, b^*, t_0^*\}$
(1) for $i = 1: n$ do
(2)   for $b = 1: 2^i$ do
(3)     OP $\longleftarrow$ matrix of all possible offloading policies
(4)     $B \longleftarrow \mathbf{OP}(b, :)$
(5)     for $k = 1: i$ do
(6)       if $(B(k) = 0)$ then
(7)         while ($a_k$execution completion) do
(8)           $D_{lk} \longleftarrow (a_k C/\nu_{mk})$ (1)
(9)           $\mathbb{E}_{lk} \longleftarrow \eta \nu_{mk}^3 D_{lk}$ (2)
(10)          $\lambda_1(k) \longleftarrow \lambda_{lk}$ (3)
(11)        end while
(12)        $t_0(k) = 0$
(13)      else
(14)        $\overline{t_0} = 0: \tau_r: 1$
(15)        for $j = 1$: length$(\overline{t_o})$ do
(16)          while ($a_k$execution completion) do
(17)            $D_{ok} \longleftarrow t_1 D_{uk}$ (11)
(18)            $\mathbb{E}_{ok} \longleftarrow D_{ok} \mathbb{P}_{mk}$ (12)
(19)            $\mathbb{E}_{wk} = t_0 D_{ok} \mu \mathbb{P}_x \alpha_d$ (17)
(20)            $\mathcal{B}_{i+1} = \mathcal{B}_i - \mathbb{E}_k + \mathbb{E}_{wk}$ (20)
(21)            $\lambda_2(j) \longleftarrow \lambda_{ok}$ (13)
(22)          end while
(23)        end for
(24)      end if
(25)      [index, $\lambda_1(k)$] = min$(\lambda_2)$
(26)      $t_0(k) \longleftarrow \overline{t_0}$(index)
(27)    end for
(28)    $\lambda_2(b) \longleftarrow$ sum$(\lambda_1)$
(29)    $t_2(b, :) \longleftarrow t_0$
(30)  end for
(31)  [index, $\lambda_3(i)$] = min$(\lambda_2)$
(32)  $\mathbf{OP}_1(i, :) \longleftarrow \mathbf{OP}$(index, : )
(33)  $t_3(i, :) \longleftarrow t_2$(index, : )
(34) end for
(35) [index, $\lambda^*$] $\longleftarrow$ min$(\lambda_3)$
(36) $t_0^* \longleftarrow t_3$(index, : )
(37) $\mathbf{OP}^* \longleftarrow \mathbf{OP}_1$(index, : )
(38) $n^* \longleftarrow$ index
(39) Save inputs: input $\longleftarrow \{a_k, \nu_{mk}, \mathbb{P}_{mk}, s_k\}$
(40) Save outputs: labels $\longleftarrow \{n^*, b^*, t_0^*\}$
(41) Repeat this for different size of applications
(42) Train the DNN: Trained_DNN $\longleftarrow$ train (input, labels)
(43) Test trained DNN
(44) Accuracy $\longleftarrow$ number of correct decisions/number of total decisions
```

Algorithm 1: DOTP algorithm.

subtasks. An offloading policy is made for the decision of subtasks to be executed either locally or remotely.

All the above techniques find the offloading policy that has minimum cost and consider time delay and energy consumption for a fixed number of subtasks per application. They also ignore the battery life of MT and optimal time allocation policy for harvesting. Our proposed DOTP considers the optimal number of subtasks per application and the optimal offloading policy. Furthermore, the DOTP also considers the current battery level of MT for improvement through harvesting energy from HAP by utilizing an optimal time allocation policy.

In Figure 4, the remaining energy of MT is plotted against various sizes of application. The battery of MT for benchmark schemes dissipates faster as compared to the proposed DOTP due to their high energy consumption. The battery level in the benchmark schemes approaches negative (practically, battery of MT does not get negative

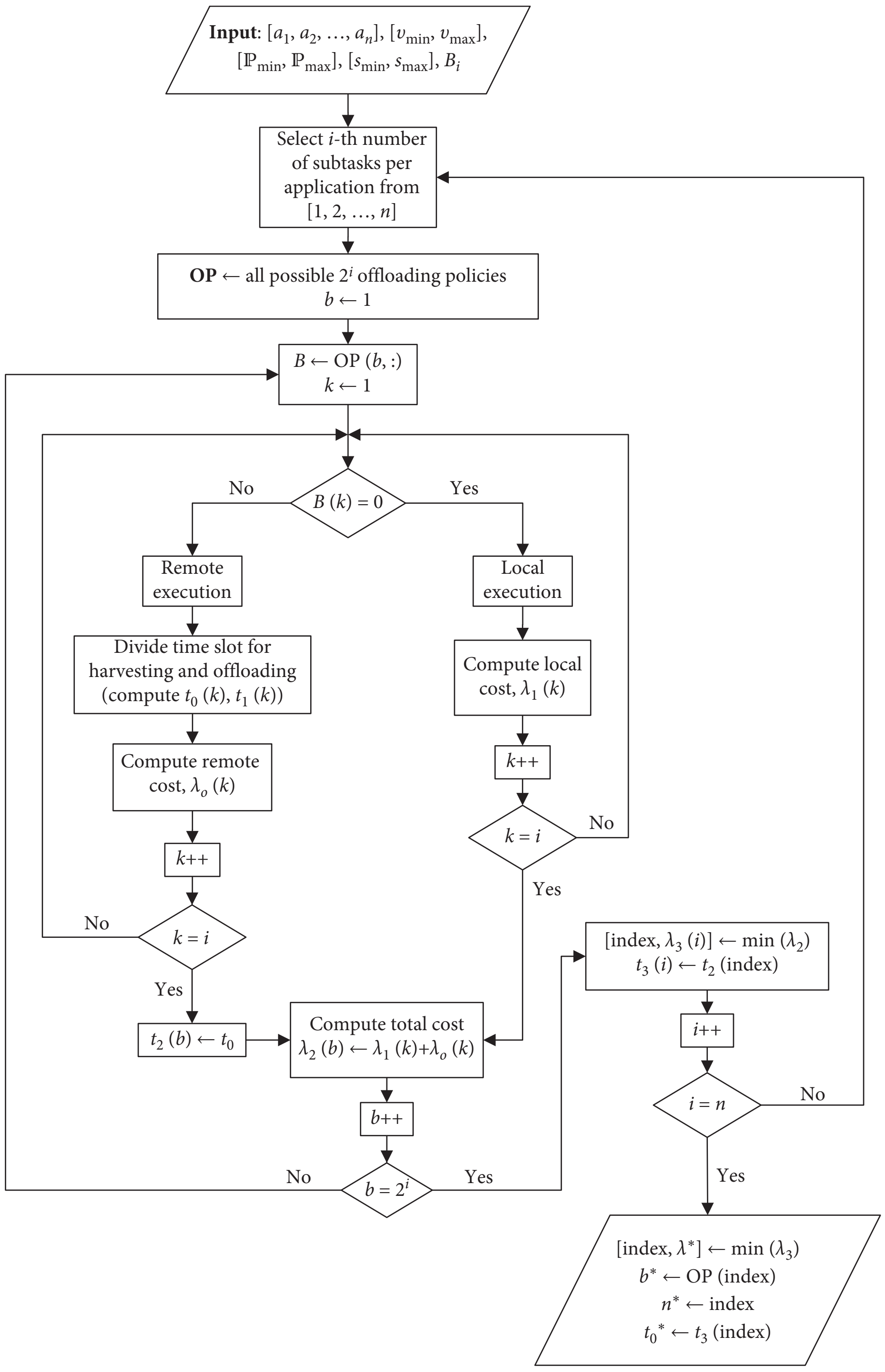
Input: $[a_1, a_2, \ldots, a_n]$, $[v_{\min}, v_{\max}]$, $[\mathbb{P}_{\min}, \mathbb{P}_{\max}]$, $[s_{\min}, s_{\max}]$, B_i
Select i-th number of subtasks per application from $[1, 2, \ldots, n]$
OP $\leftarrow$ all possible 2^i offloading policies $b \leftarrow 1$
$B \leftarrow$ OP $(b, :)$ $k \leftarrow 1$
$B(k) = 0$
No
Yes
Remote execution
Local execution
Divide time slot for harvesting and offloading (compute $t_0(k)$, $t_1(k)$)
Compute local cost, $\lambda_1(k)$
Compute remote cost, $\lambda_o(k)$
$k{+}{+}$
$k = i$
No
Yes
$t_2(b) \leftarrow t_0$
Compute total cost $\lambda_2(b) \leftarrow \lambda_1(k) + \lambda_o(k)$
$b{+}{+}$
$b = 2^i$
[index, $\lambda_3(i)$] $\leftarrow$ min (λ_2) $t_3(i) \leftarrow t_2$ (index)
$i{+}{+}$
$i = n$
[index, λ^*] $\leftarrow$ min (λ_3)
$b^* \leftarrow$ OP (index)
$n^* \leftarrow$ index
$t_0{}^* \leftarrow t_3$ (index)

Figure 2: Flowchart of proposed DOTP algorithm.

values, and it is possible only in the simulations; in simulation results, negative values for MT's battery denote the shutdown state of MT) that results in the shutdown of MT, and consequently disconnection with HAP occurs. The total remaining energy for DOTP increases after a task size of 12 GBs because the harvesting energy time fraction of MT is dependent upon the remaining battery of MT. As the battery of MT decreases below a threshold level, the MT increases the time for harvesting energy and decreases the time for data offloading by changing the weighting coefficients' values described in the mathematical model. In contrast, there is no intelligent division of time slot for energy harvesting and data offloading in the benchmark schemes. Therefore, the remaining energy of MT for other schemes further decreases after task size of 12 GBs. Thus, in the DOTP scheme, a reserved battery level always remains available in the MT, which ensures that neither the MT will be

Table 5: Simulation parameters [15].

Parameter	Value
β	0.5 MHz
n	$\{1, 2, 3, \ldots, 10\}$
C	737.5 cycles/byte
s	$[3\ \ 200]$ m
ν_m	$[0.1\ \ 1]$ GHz
η	10^{-25}
$\mathbb{P}_m$	$[0.8\ \ 1.5]$ Watt
N_o	−174 dBm/Hz
M	$[0.1\ \ 1]$ G bits
μ	0.8
ρ	−30 dB
s_0	1 m
t_0	$[0\ \ 1]$
τ_r	0.01
ϕ	≥ 2

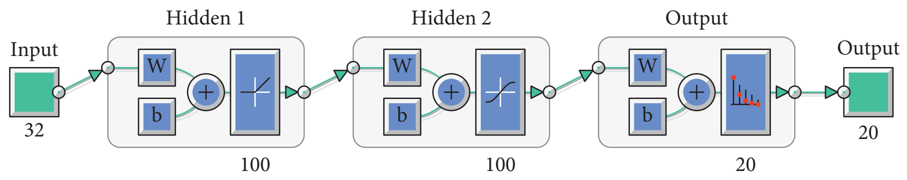

Figure 3: Detailed architecture of DNN.

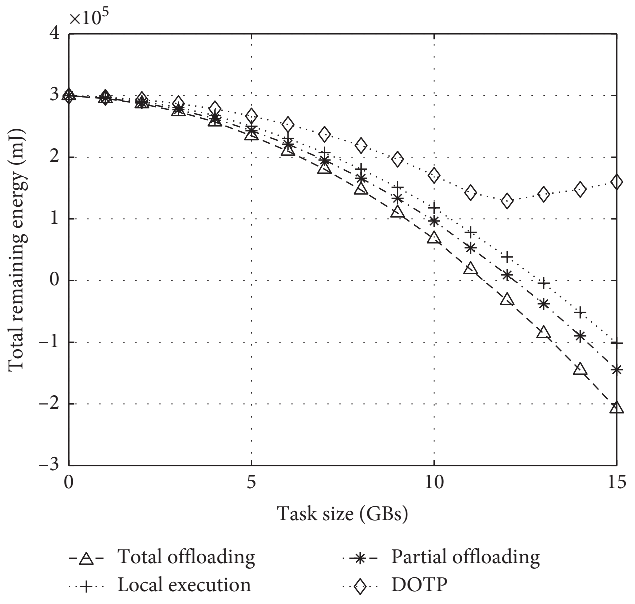

Figure 4: Battery life of MT with varying application sizes.

disconnected from HAP nor the battery level will go to negative values.

Figure 5 depicts the plot of total energy consumption with varying application sizes for all the mentioned techniques. The minimum energy consumption of DOTP can be observed from the figure. This is due to the consideration of different energy consumptions in the cost function (ignored in the benchmark schemes) and then the selection of offloading policy based on optimal number of subtasks. The different energy consumption behavior of first increasing and then decreasing when a threshold level is reached is due to the reason that the amount of energy being harvested is greater than that of the consumed energy. As MT prefers energy harvesting, the total energy consumption decreases at the task size of 12 GB. It is due to the energy consumption for comparatively smaller time fraction in the data offloading, while the energy consumption for the other schemes continuously increases due to the absence of smart time allocation policy. To the best of our knowledge, this smart time allocation policy is ignored in all the considered benchmark techniques.

Figure 6 compares the time delay versus different application sizes plot of DOTP with the benchmark schemes. The better performance of the proposed DOTP is evident from the figure through achievement of smaller time delays. This better performance of DOTP is due to the selection of optimal number of subtasks per application, which is considered constant in the mentioned benchmark schemes. The total offloading has minimum time delay, but this technique's energy consumption is the highest, as depicted in Figure 5. The greater rise in the execution time of DOTP is due to the lower battery level of MT; that is, most of the time the MT will harvest energy to level up its battery and in the same time slot little offloading of data takes place. The MT harvests energy for greater part of a time slot; therefore, an application will be offloaded in more time.

In Figure 7, the cost of the proposed DOTP is compared with those of the three benchmark schemes. From the figure, it is evident that the DOTP has the lowest cost as compared

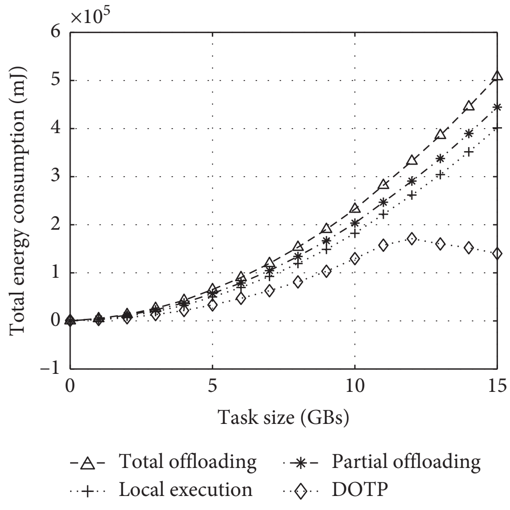

Figure 5: Total energy consumption for various application sizes.

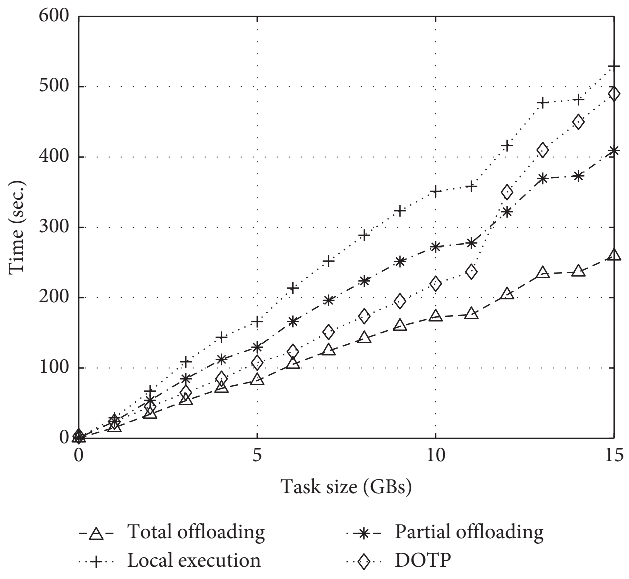

Figure 6: Time delay of MT with varying application sizes.

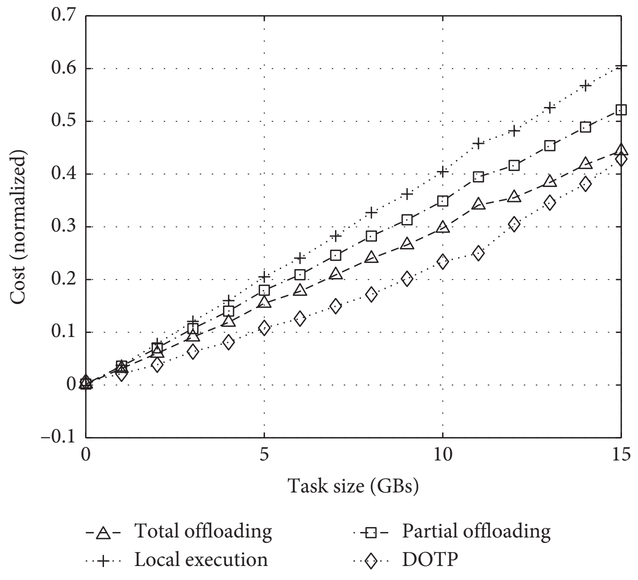

Figure 7: Cost for different application sizes.

to the other techniques. This minimum cost is the result of selecting optimal number of subtasks per application and considering the energy consumption, time delay, and energy harvesting in the cost function. In the cost function, weighting coefficients are dependent on the remaining energy of MT. These coefficients are used for providing priority to either data offloading or energy harvesting of MT. In contrast, the benchmark schemes consider fixed number of subtasks per application and focus on either energy consumption or time delay but not both simultaneously. The minimum cost of DOTP demonstrates the better performance for computational offloading in terms of energy consumption, time delay, and battery life.

5. Conclusions

In this paper, the battery life of MT, time allocation policy for energy harvesting and data offloading, and offloading policy based on optimal number of subtasks are investigated for single MT in WP-MEC system. The application for computational offloading is divided into optimal number of subtasks and then, through offloading policy, the decision to follow either remote execution or local execution is made. A detailed cost function is formulated, which jointly considers time delay and energy consumption for offloading policy. In addition, the cost function includes the current battery level of MT and thus prolongs the battery life of MT. The deep learning approach is used to find the optimal number of subtasks per application and partial offloading policy along with time allocation policy for energy harvesting simultaneously. The DOTP algorithm provides time allocation policy that results in extended battery life as compared to the benchmark schemes. A DNN is trained on a pregenerated dataset through mathematical model for the achievement of aforementioned objectives in order to reduce the energy consumption and time delay that incur for traditional optimization methods and make the decisions faster. It is evident from the numerical results that our proposed DOTP demonstrates extended battery life, lower energy consumption, and lower time delay for application execution in comparison with the benchmark techniques. In the future, the scenario can be extended to multiple MTs and HAPs, as well as to vehicular networks. In addition, reinforcement approach can be used, and routing strategies can be investigated for handover management during mobility from one HAP to another.

Data Availability

The data used in this study are random and are generated through our proposed mathematical model and Algorithm 1. Using the simulation parameters, given in Table 5, the required data can be generated as explained in Section 3.3.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] A. Ahad, M. Tahir, M. Aman Sheikh, K. I. Ahmed, A. Mughees, and A. Numani, "Technologies trend towards 5G network for smart health-care using IoT: a review," *Sensors*, vol. 20, no. 14, p. 4047, 2020.

[2] Z. Ali, Z. H. Abbas, and F. Y. Li, "A stochastic routing algorithm for distributed IoT with unreliable wireless links," in *Proceedings of the 2016 IEEE 83rd Vehicular Technology Conference (VTC Spring)*, pp. 1–5, IEEE, Nanjing, China, May 2016.

[3] A. Shakarami, M. Ghobaei-Arani, and A. Shahidinejad, "A survey on the computation offloading approaches in mobile edge computing: a machine learning-based perspective," *Computer Networks*, vol. 182, p. 107496, 2020.

[4] Z. Ali, S. Khaf, Z. H. Abbas, G. Abbas, F. Muhammad, and S. Kim, "A deep learning approach for mobility-aware and energy-efficient resource allocation in mec," *IEEE Access*, vol. 8, pp. 179530–179546, 2020.

[5] X. Jin, Z. Wang, and W. Hua, "Cooperative runtime offloading decision algorithm for mobile cloud computing," *Mobile Information Systems*, vol. 2019, Article ID 8049804, 17 pages, 2019.

[6] J. Ren, G. Yu, Y. He, and G. Y. Li, "Collaborative cloud and edge computing for latency minimization," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 5, pp. 5031–5044, 2019.

[7] Z. Ali, S. Khaf, Z. H. Abba, G. Abbas et al., "A comprehensive utility function for resource allocation in mobile edge computing," *Computers, Materials & Continua*, vol. 66, no. 2, pp. 1461–1477, 2020.

[8] S. P. Singh, A. Nayyar, R. Kumar, and A. Sharma, "Fog computing: from architecture to edge computing and big data processing," *The Journal of Supercomputing*, vol. 75, no. 4, pp. 2070–2105, 2019.

[9] F. O. Ombongi, H. O. Absaloms, and P. L. Kibet, "Resource allocation in millimeter-wave device-to-device networks," *Mobile Information Systems*, vol. 2019, Article ID 5051360, 16 pages, 2019.

[10] O. L. López, H. Alves, R. D. Souza, S. Montejo-Sánchez, E. M. Fernandez, and M. Latva-Aho, "Massive wireless energy transfer: enabling sustainable IoT towards 6G era," *IEEE Internet of Things Journal*, vol. 8, no. 11, pp. 8816–8835, 2021.

[11] D. D. Tran, D. B. Ha, and A. Nayyar, "Wireless power transfer under secure communication with multiple antennas and eavesdroppers," in *Proceedings of the International Conference on Industrial Networks and Intelligent Systems*, pp. 208–220, Da Nang, Vietnam, August 2018.

[12] S. Bi and Y. J. Zhang, "Computation rate maximization for wireless powered mobile-edge computing with binary computation offloading," *IEEE Transactions on Wireless Communications*, vol. 17, no. 6, pp. 4177–4190, 2018.

[13] H. Eom, R. Figueiredo, H. Cai, Y. Zhang, and G. Huang, "Malmos: machine learning-based mobile offloading scheduler with online training," in *Proceedings of the 2015 3rd IEEE International Conference on Mobile Cloud Computing, Services, and Engineering*, pp. 51–60, San Francisco, CA, USA, March 2015.

[14] Z. Kuang, L. Li, J. Gao, L. Zhao, and A. Liu, "Partial offloading scheduling and power allocation for mobile edge computing systems," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 6774–6785, 2019.

[15] C. Li, W. Chen, H. Tang, Y. Xin, and Y. Luo, "Stochastic computation resource allocation for mobile edge computing powered by wireless energy transfer," *Ad Hoc Networks*, vol. 93, p. 101897, 2019.

[16] A. Irshad, Z. H. Abbas, Z. Ali, G. Abbas, T. Baker, and D. Al-Jumeily, "Wireless powered mobile edge computing systems: simultaneous time allocation and offloading policies," *Electronics*, vol. 10, no. 8, p. 965, 2021.

[17] F. Zhou, Y. Wu, R. Q. Hu, and Y. Qian, "Computation rate maximization in UAV-enabled wireless-powered mobile-edge computing systems," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 9, pp. 1927–1941, 2018.

[18] F. Wang, J. Xu, X. Wang, and S. Cui, "Joint offloading and computing optimization in wireless powered mobile-edge computing systems," *IEEE Transactions on Wireless Communications*, vol. 17, no. 3, pp. 1784–1797, 2017.

[19] D. Wu, F. Wang, X. Cao, and J. Xu, "Wireless powered user cooperative computation in mobile edge computing systems," in *Proceedings of the 2018 IEEE GLOBECOM Workshops (GC Wkshps)*, pp. 1–7, Abu Dhabi, UAE, December 2018.

[20] S. Mao, S. Leng, K. Yang, X. Huang, and Q. Zhao, "Fair energy-efficient scheduling in wireless powered full-duplex mobile-edge computing systems," in *Proceedings of the GLOBECOM 2017-2017 IEEE Global Communications Conference*, pp. 1–6, Singapore, December 2017.

[21] A. Mahmood, A. Ahmed, M. Naeem, and Y. Hong, "Partial offloading in energy harvested mobile edge computing: a direct search approach," *IEEE Access*, vol. 8, pp. 36757–36763, 2020.

[22] C. Li, M. Song, L. Zhang, W. Chen, and Y. Luo, "Offloading optimization and time allocation for multiuser wireless energy transfer based mobile edge computing system," *Mobile Networks and Applications*, vol. 25, pp. 1–9, 2020.

[23] D. Zhang, Z. Chen, M. K. Awad, N. Zhang, H. Zhou, and X. S. Shen, "Utility-optimal resource management and allocation algorithm for energy harvesting cognitive radio sensor networks," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 12, pp. 3552–3565, 2016.

[24] S. Mao, S. Leng, S. Maharjan, and Y. Zhang, "Energy efficiency and delay tradeoff for wireless powered mobile-edge computing systems with multi-access schemes," *IEEE Transactions on Wireless Communications*, vol. 19, no. 3, pp. 1855–1867, 2019.

[25] L. Huang, S. Bi, and Y.-J. A. Zhang, "Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks," *IEEE Transactions on Mobile Computing*, vol. 19, no. 11, pp. 2581–2593, 2019.

[26] Z. Ali, L. Jiao, T. Baker, G. Abbas, Z. H. Abbas, and S. Khaf, "A deep learning approach for energy efficient computational offloading in mobile edge computing," *IEEE Access*, vol. 7, pp. 149623–149633, 2019.

[27] Z. H. Abbas, Z. Ali, G. Abbas et al., "Computational offloading in mobile edge with comprehensive and energy efficient cost function: a deep learning approach," *Sensors*, vol. 21, no. 10, p. 3523, 2021.

[28] T. D. Burd and R. W. Brodersen, "Processor design for portable systems," *Journal of VLSI Signal Processing Systems for Signal, Image and Video Technology*, vol. 13, no. 2, pp. 203–221, 1996.

[29] Y. Mao, J. Zhang, S. Song, and K. B. Letaief, "Power-delay tradeoff in multi-user mobile-edge computing systems," in *Proceedings of the 2016 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, Washington, DC, USA, December 2016.

[30] Y. G. Kim, M. Kim, and S. W. Chung, "Enhancing energy efficiency of multimedia applications in heterogeneous mobile multi-core processors," *IEEE Transactions on Computers*, vol. 66, no. 11, pp. 1878–1889, 2017.